FIG. 1

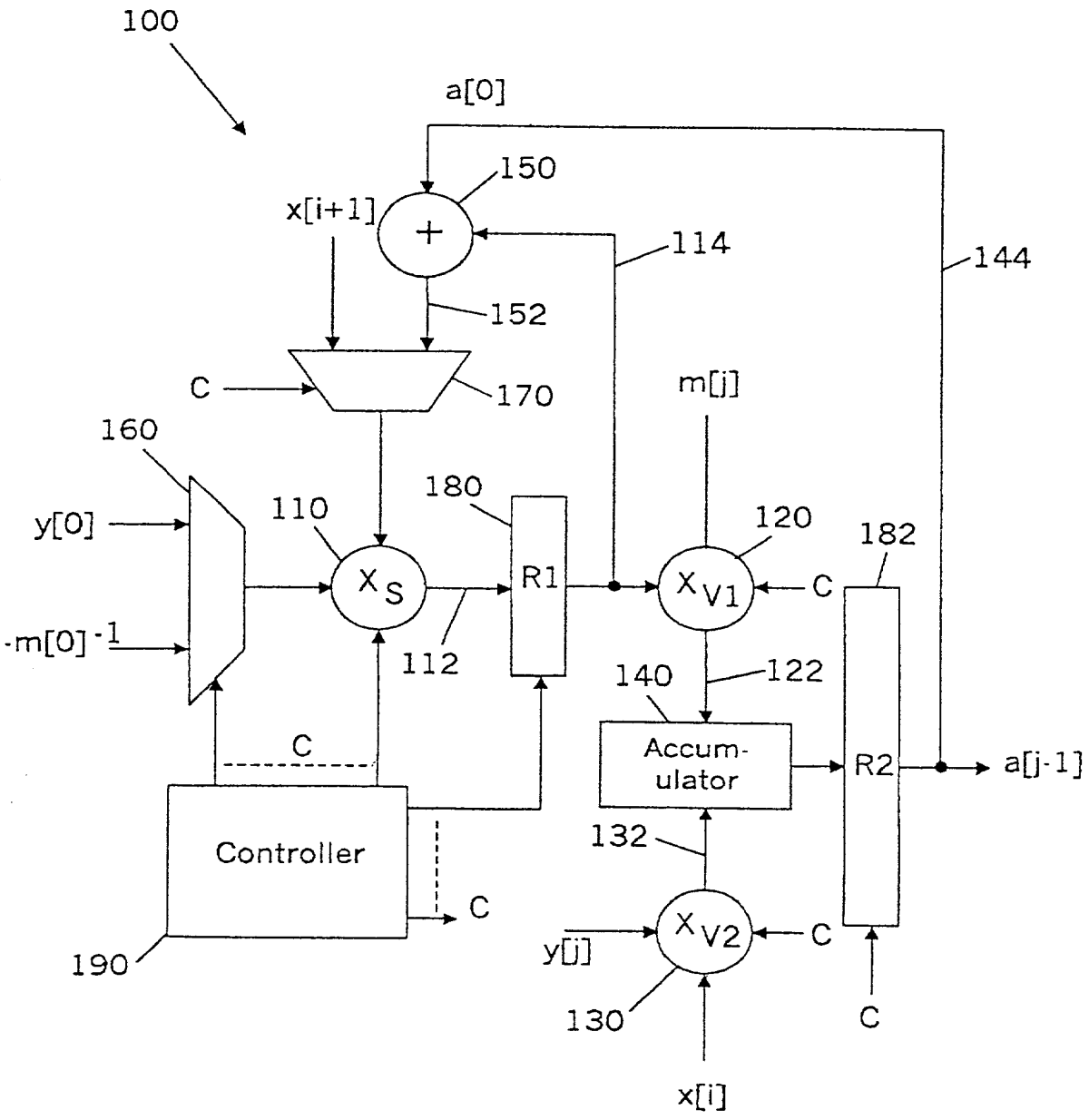

100
a[0]
150
x[i+1]
+
114
144
152
C
170
m[j]
160
180
110
120
182
y[0]
X_S
R1
X_{V1}
C
-m[0]
-1
112
140
122
C
Accum-
ulator
R2
a[j-1]
Controller
132
C
X_{V2}
C
y[j]
130
C
190
x[i]

FIG. 2

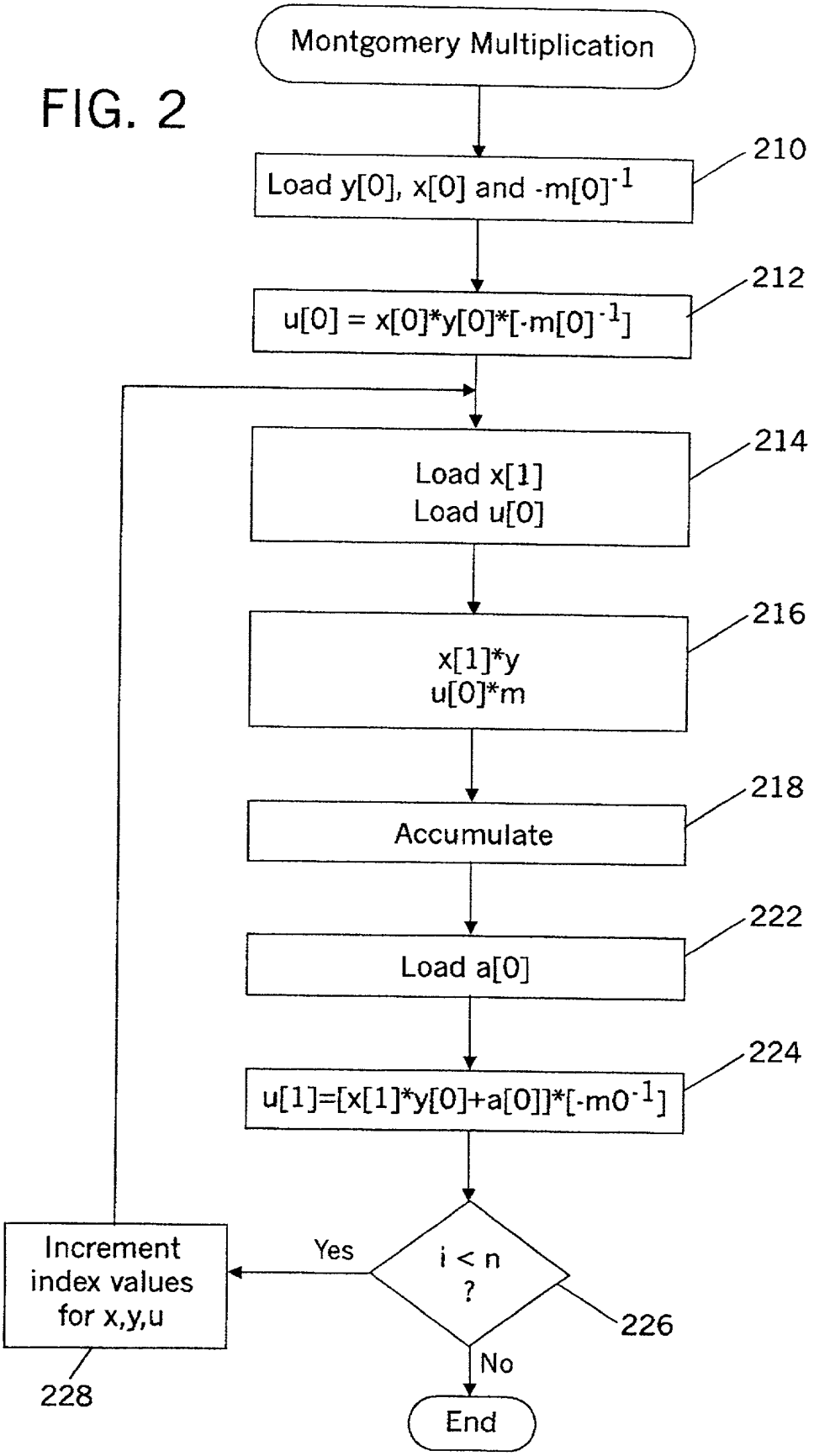
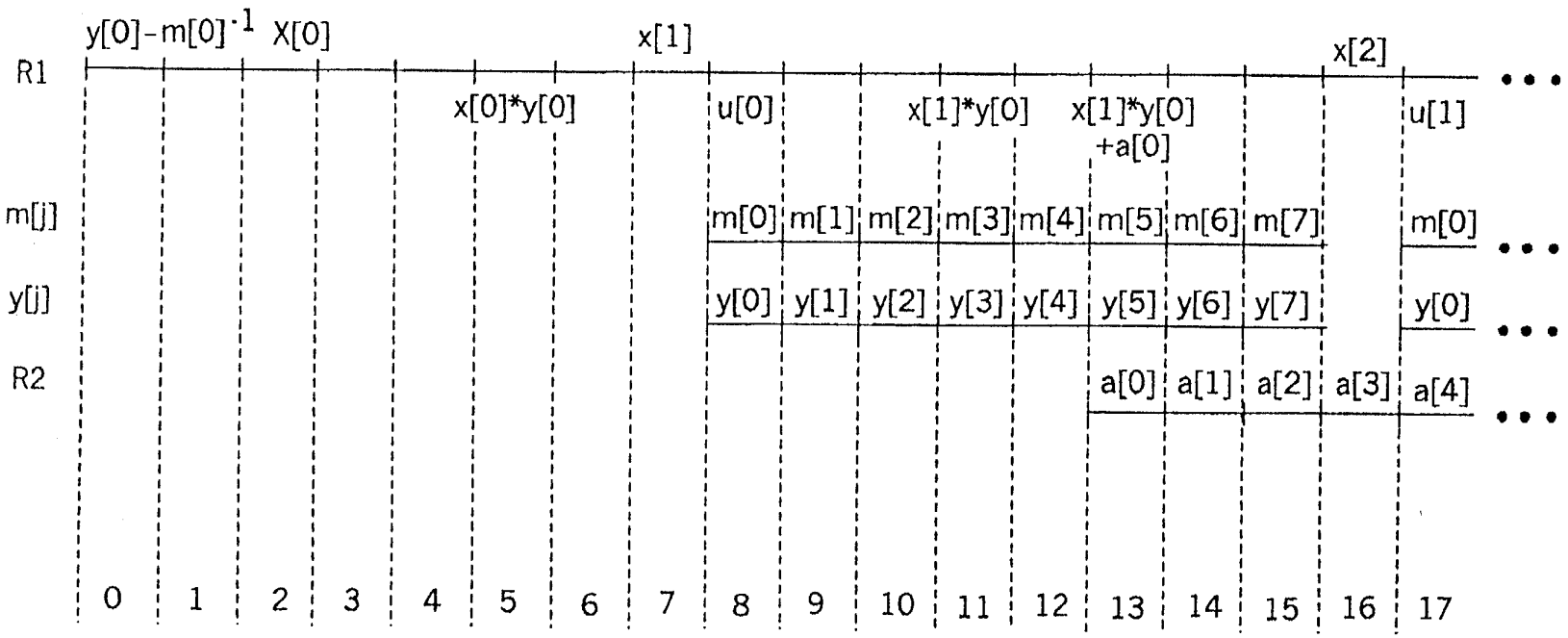

Montgomery Multiplication
Load y[0], x[0] and -m[0]-1
210
u[0] = x[0]*y[0]*[-m[0]-1]
212
Load x[1]
Load u[0]
214
x[1]*y
u[0]*m
216
Accumulate
218
Load a[0]
222
u[1]=[x[1]*y[0]+a[0]]*[-m0-1]
224
i < n
?
226
Yes
No
Increment
index values
for x,y,u
228
End

FIG. 3



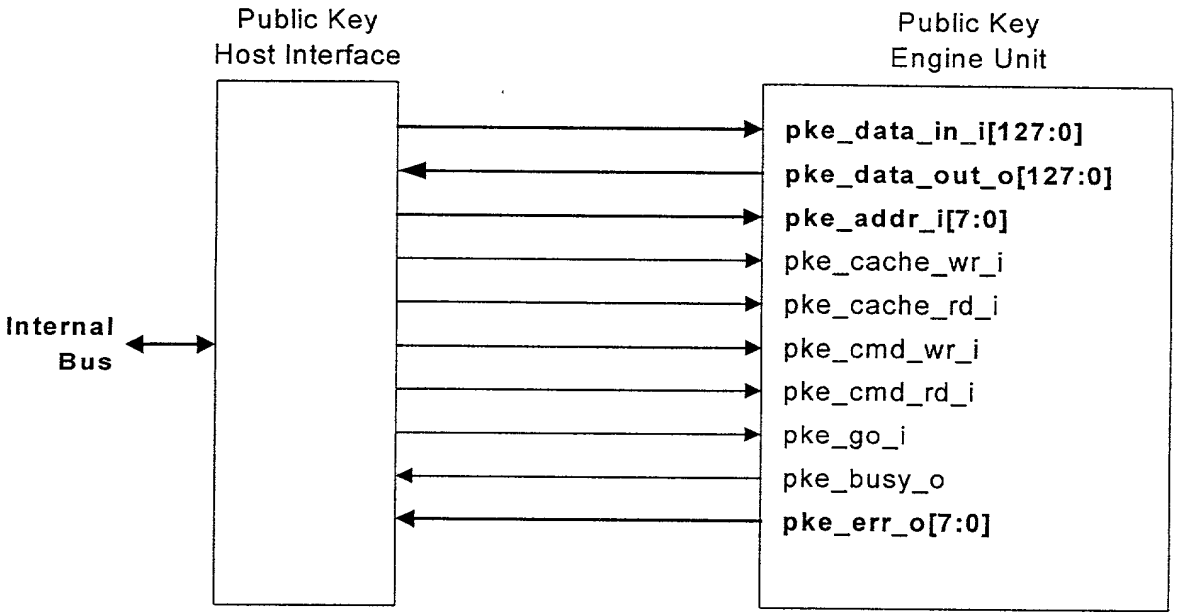

Public Key
Host Interface
Public Key
Engine Unit
Internal
Bus
pke_data_in_i[127:0]
pke_data_out_o[127:0]
pke_addr_i[7:0]
pke_cache_wr_i
pke_cache_rd_i
pke_cmd_wr_i
pke_cmd_rd_i
pke_go_i
pke_busy_o
pke_err_o[7:0]

FIG. 4

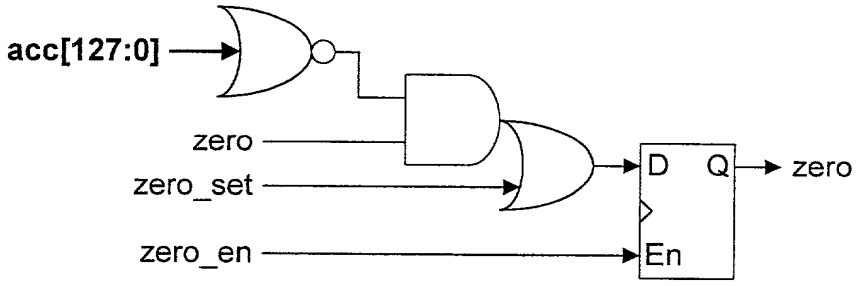

acc[127:0]
zero
zero_set
zero_en
D
Q
En
zero

FIG. 12

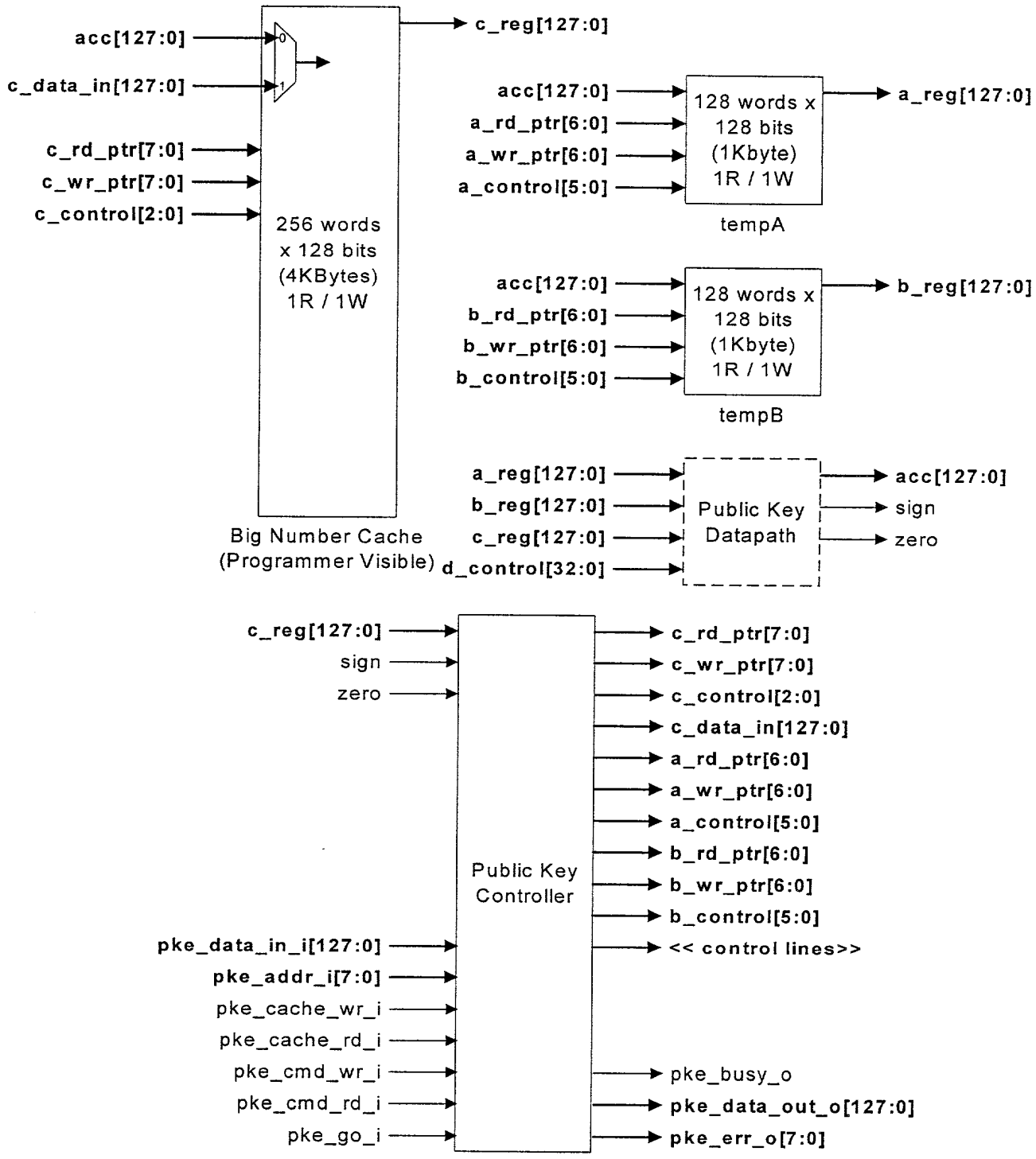

acc[127:0]
c_data_in[127:0]
c_rd_ptr[7:0]
c_wr_ptr[7:0]
c_control[2:0]
0
1
256 words
x 128 bits
(4KBytes)
1R / 1W
Big Number Cache
(Programmer Visible)
c_reg[127:0]
acc[127:0]
a_rd_ptr[6:0]
a_wr_ptr[6:0]
a_control[5:0]
128 words x
128 bits
(1Kbyte)
1R / 1W
tempA
a_reg[127:0]
acc[127:0]
b_rd_ptr[6:0]
b_wr_ptr[6:0]
b_control[5:0]
128 words x
128 bits
(1Kbyte)
1R / 1W
tempB
b_reg[127:0]
a_reg[127:0]
b_reg[127:0]
c_reg[127:0]
d_control[32:0]
Public Key
Datapath
acc[127:0]
sign
zero
c_reg[127:0]
sign
zero
Public Key
Controller
c_rd_ptr[7:0]
c_wr_ptr[7:0]
c_control[2:0]
c_data_in[127:0]
a_rd_ptr[6:0]
a_wr_ptr[6:0]
a_control[5:0]
b_rd_ptr[6:0]
b_wr_ptr[6:0]
b_control[5:0]
<< control lines>>
pke_data_in_i[127:0]
pke_addr_i[7:0]
pke_cache_wr_i
pke_cache_rd_i
pke_cmd_wr_i
pke_cmd_rd_i
pke_go_i
pke_busy_o
pke_data_out_o[127:0]
pke_err_o[7:0]

FIG. 5

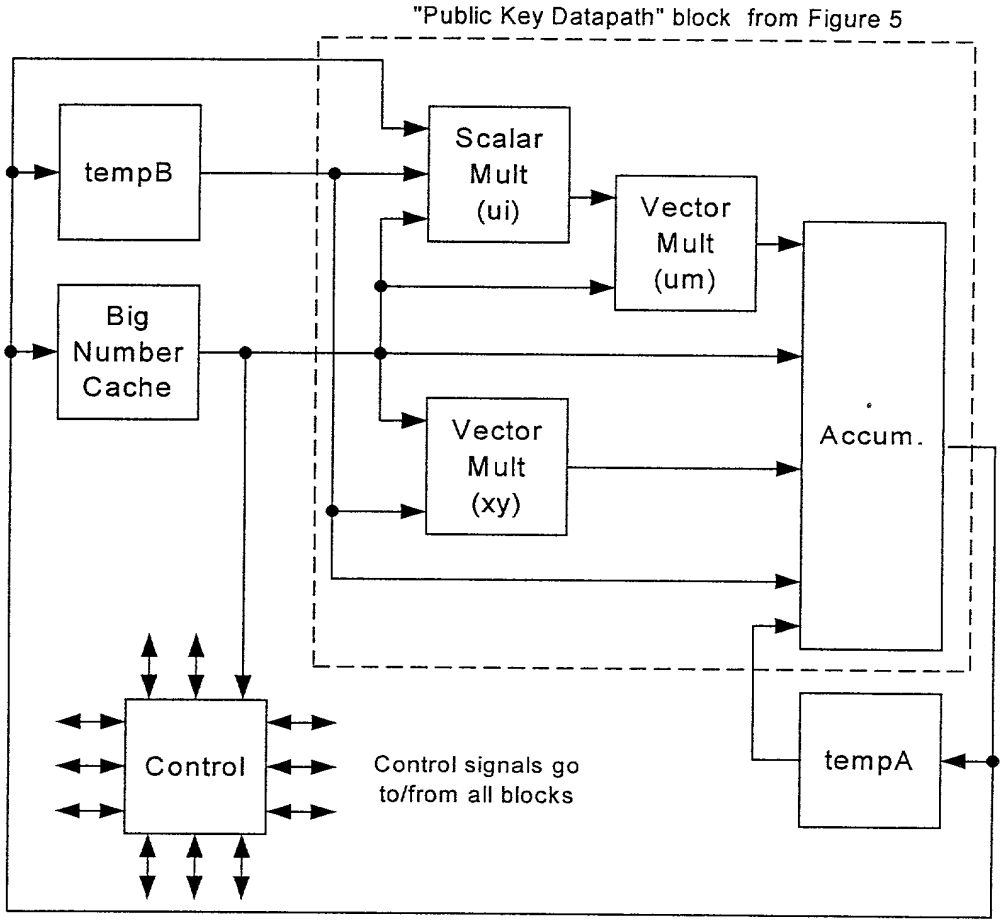

"Public Key Datapath" block from Figure 5
tempB
Big Number Cache
Scalar Mult (ui)
Vector Mult (um)
Vector Mult (xy)
Accum.
tempA
Control
Control signals go to/from all blocks

FIG. 6

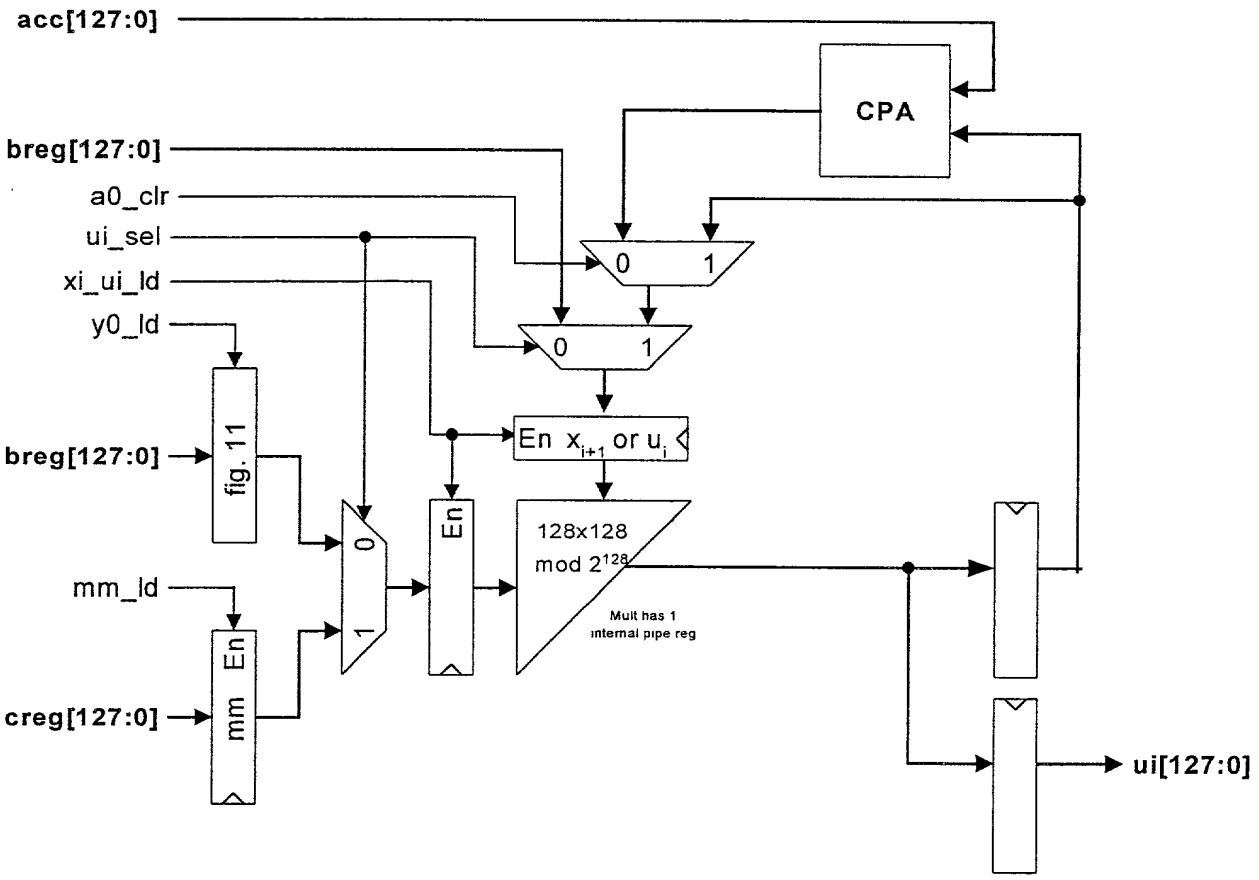

acc[127:0]
CPA
breg[127:0]
a0_clr
ui_sel
xi_ui_ld
y0_ld
0
1
0
1
fig. 11
breg[127:0]
En x$_{i+1}$ or u$_i$
0
1
En
128x128
mod 2^{128}
Mult has 1
internal pipe reg
mm_ld
mm En
creg[127:0]
ui[127:0]

FIG. 7

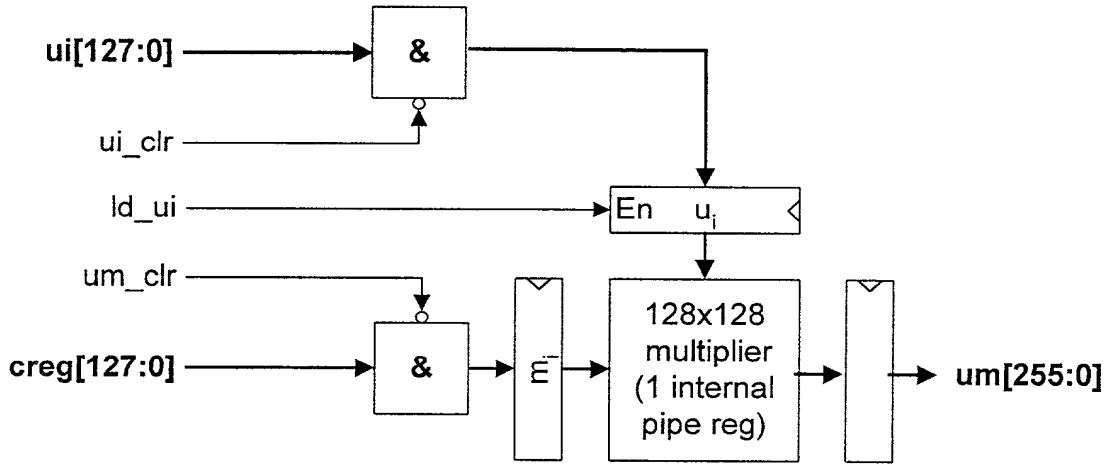

ui[127:0]
&
ui_clr
ld_ui
En u_i
um_clr
creg[127:0]
&
128x128
multiplier
(1 internal
pipe reg)
um[255:0]

FIG. 8

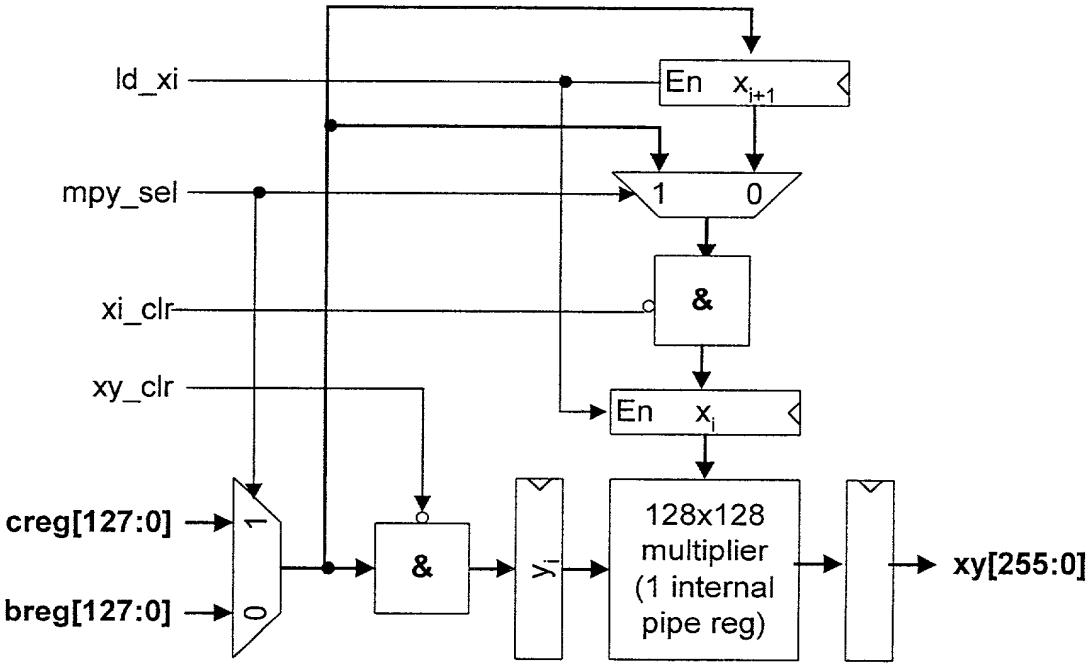
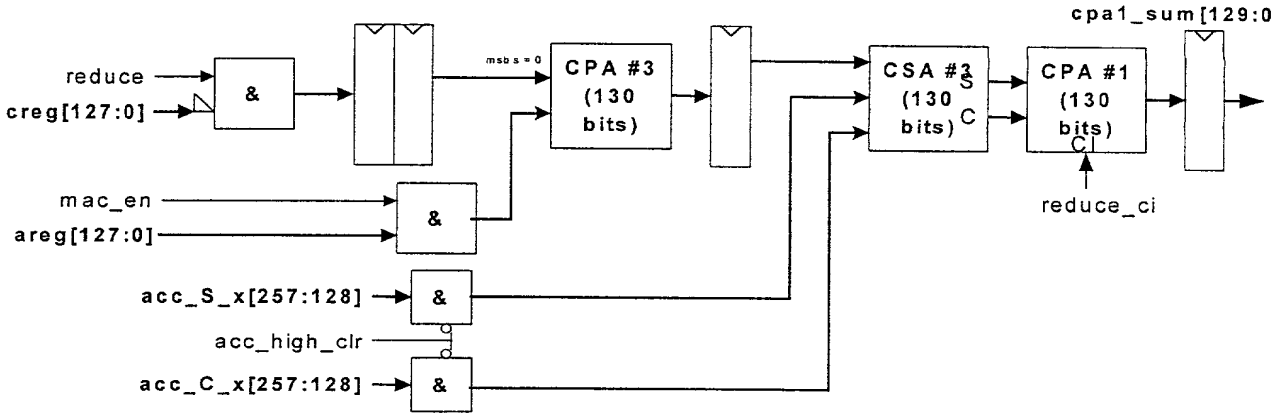

ld_xi
En x_i+1
mpy_sel
1
0
xi_clr
&
xy_clr
En x_i
creg[127:0]
breg[127:0]
1
0
&
128x128
multiplier
(1 internal
pipe reg)
xy[255:0]

FIG. 9


reduce
creg[127:0]
&
msbs = 0
CPA #3
(130
bits)
CSA #3
(130
bits)
S
C
CPA #1
(130
bits)
C
cpa1_sum[129:0
reduce_ci
mac_en
areg[127:0]
&
acc_S_x[257:128]
&
acc_high_clr
acc_C_x[257:128]
&

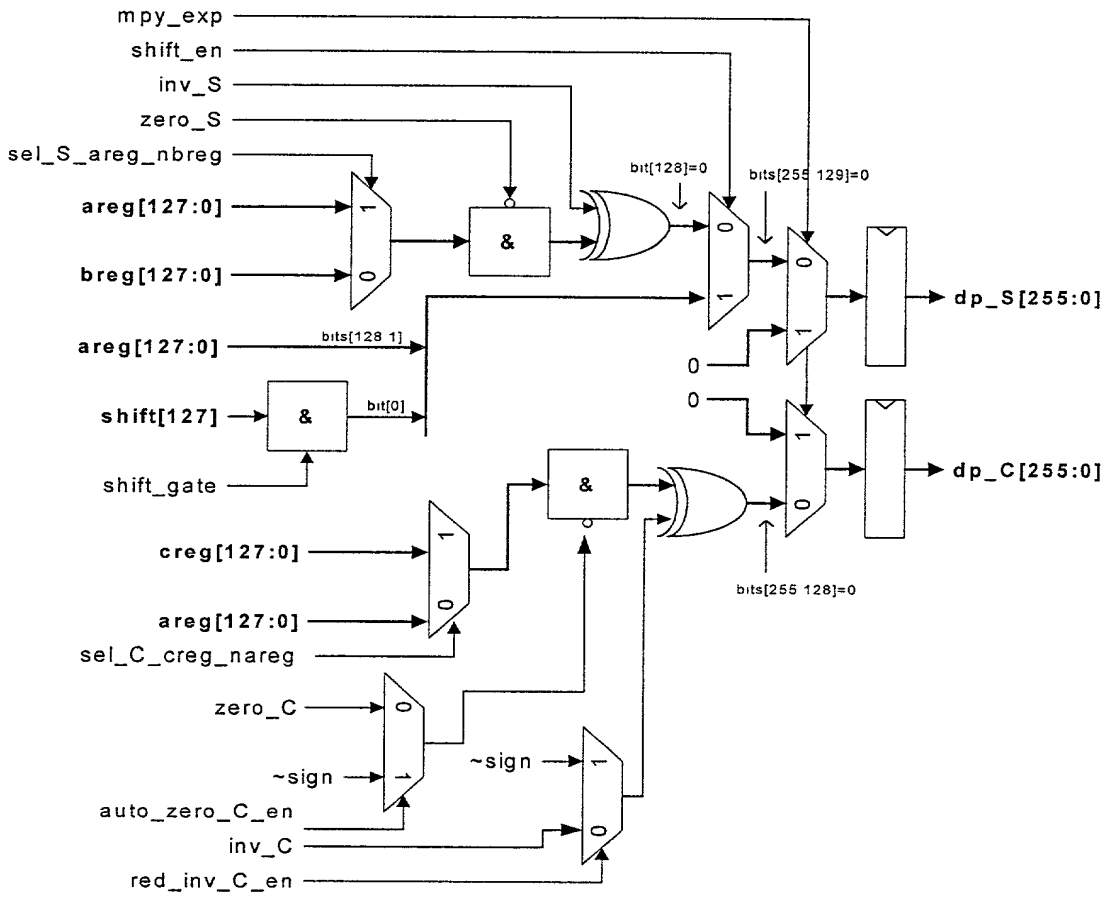

mpy_exp
shift_en
inv_S
zero_S
sel_S_areg_nbreg
areg[127:0]
breg[127:0]
&
bit[128]=0
bits[255 129]=0
dp_S[255:0]
areg[127:0]
bits[128 1]
0
shift[127]
&
bit[0]
0
shift_gate
&
dp_C[255:0]
creg[127:0]
areg[127:0]
bits[255 128]=0
sel_C_creg_nareg
zero_C
~sign
~sign
auto_zero_C_en
inv_C
red_inv_C_en

FIG. 10

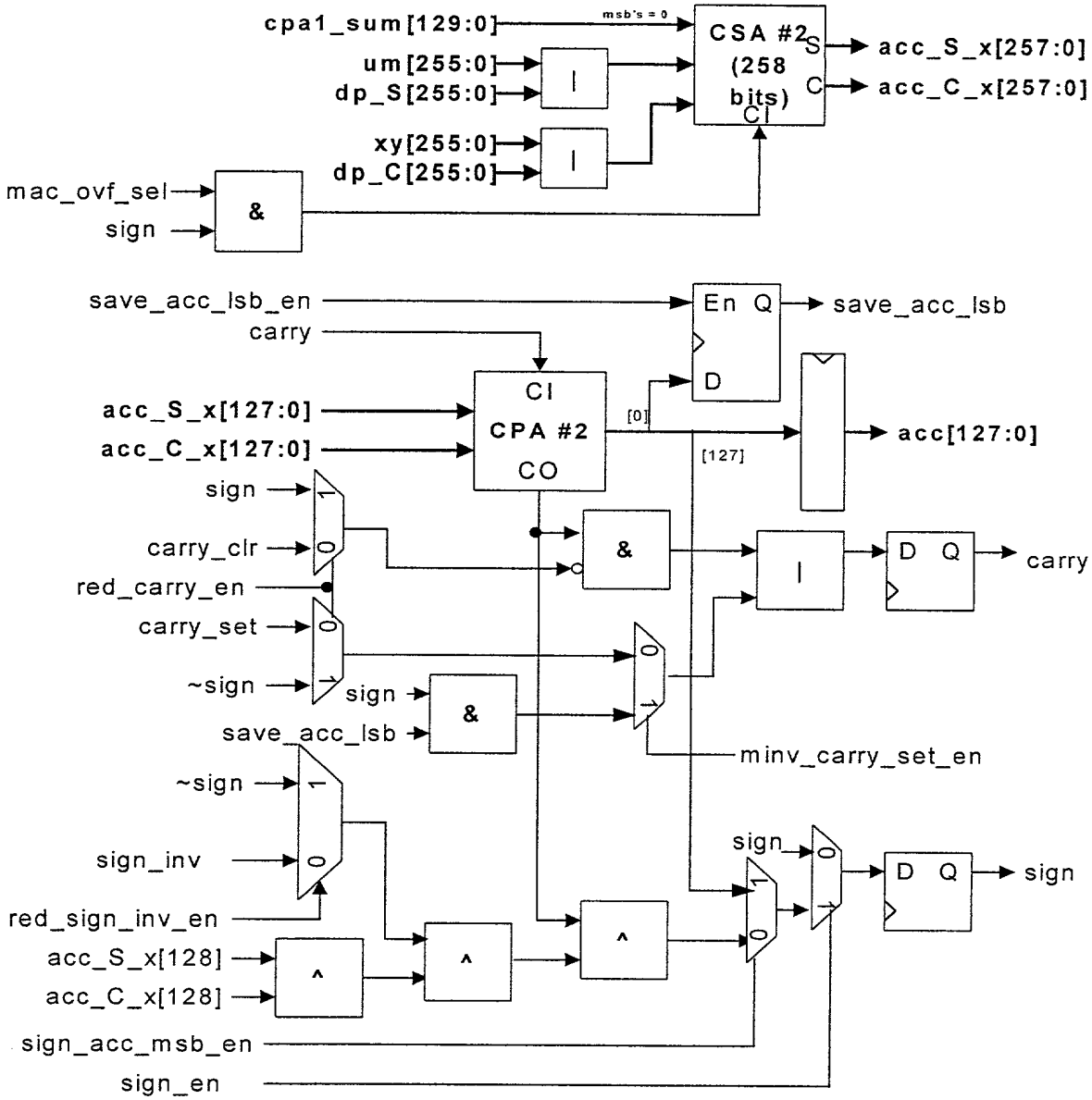

cpa1_sum[129:0]
msb's = 0
CSA #2 (258 bits)
S
C
CI
acc_S_x[257:0]
acc_C_x[257:0]
um[255:0]
dp_S[255:0]
xy[255:0]
dp_C[255:0]
mac_ovf_sel
sign
&
save_acc_lsb_en
carry
En Q
D
save_acc_lsb
CI
CPA #2
CO
acc_S_x[127:0]
acc_C_x[127:0]
[0]
[127]
acc[127:0]
sign
carry_clr
red_carry_en
carry_set
~sign
sign
save_acc_lsb
&
|
D Q
carry
minv_carry_set_en
~sign
sign_inv
red_sign_inv_en
acc_S_x[128]
acc_C_x[128]
^
sign
sign_acc_msb_en
sign_en

FIG. 11

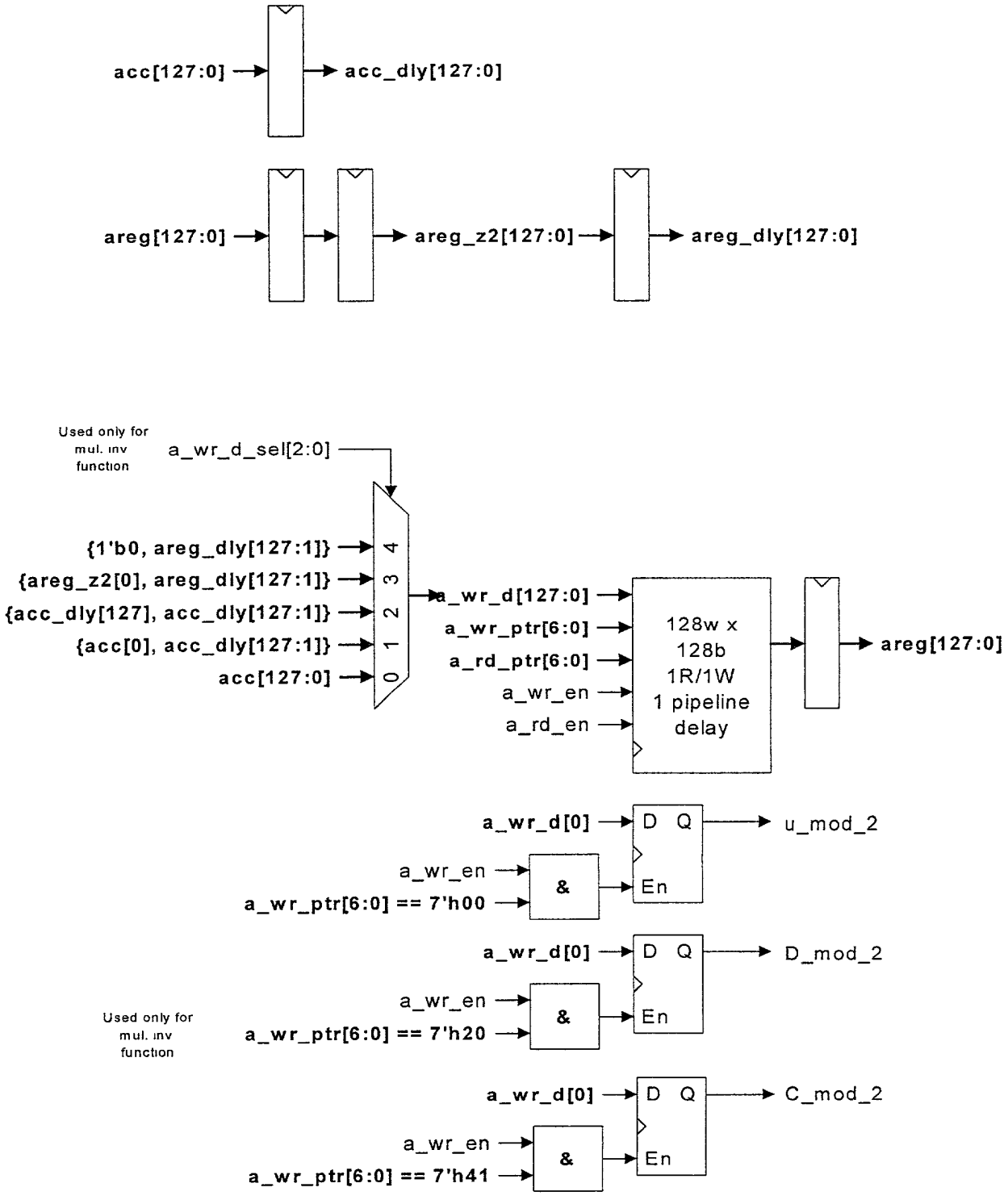

acc[127:0]
acc_dly[127:0]
areg[127:0]
areg_z2[127:0]
areg_dly[127:0]
Used only for mul. inv function
a_wr_d_sel[2:0]
{1'b0, areg_dly[127:1]}
{areg_z2[0], areg_dly[127:1]}
{acc_dly[127], acc_dly[127:1]}
{acc[0], acc_dly[127:1]}
acc[127:0]
0 1 2 3 4
a_wr_d[127:0]
a_wr_ptr[6:0]
a_rd_ptr[6:0]
a_wr_en
a_rd_en
128w x 128b 1R/1W 1 pipeline delay
areg[127:0]
a_wr_d[0]
D Q
u_mod_2
a_wr_en
a_wr_ptr[6:0] == 7'h00
&
En
a_wr_d[0]
D Q
D_mod_2
a_wr_en
a_wr_ptr[6:0] == 7'h20
&
En
Used only for mul. inv function
a_wr_d[0]
D Q
C_mod_2
a_wr_en
a_wr_ptr[6:0] == 7'h41
&
En

FIG. 13

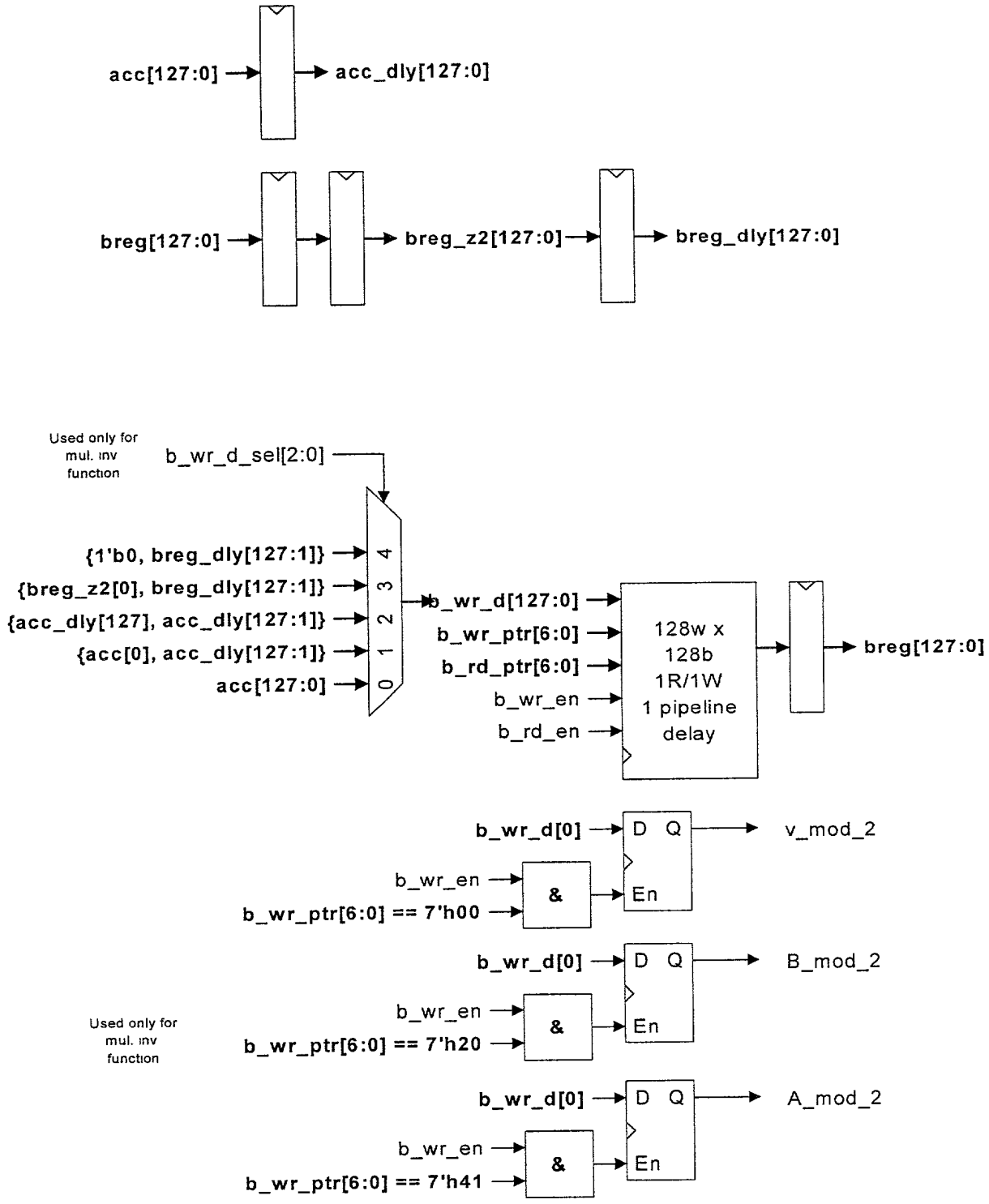

acc[127:0]
acc_dly[127:0]
breg[127:0]
breg_z2[127:0]
breg_dly[127:0]
Used only for mul. inv function
b_wr_d_sel[2:0]
{1'b0, breg_dly[127:1]}
{breg_z2[0], breg_dly[127:1]}
{acc_dly[127], acc_dly[127:1]}
{acc[0], acc_dly[127:1]}
acc[127:0]
0 1 2 3 4
b_wr_d[127:0]
b_wr_ptr[6:0]
b_rd_ptr[6:0]
b_wr_en
b_rd_en
128w x 128b 1R/1W 1 pipeline delay
breg[127:0]
b_wr_d[0]
D Q
v_mod_2
b_wr_en
b_wr_ptr[6:0] == 7'h00
&
En
B_mod_2
b_wr_ptr[6:0] == 7'h20
A_mod_2
b_wr_ptr[6:0] == 7'h41
Used only for mul. inv function

FIG. 14

FIG. 15

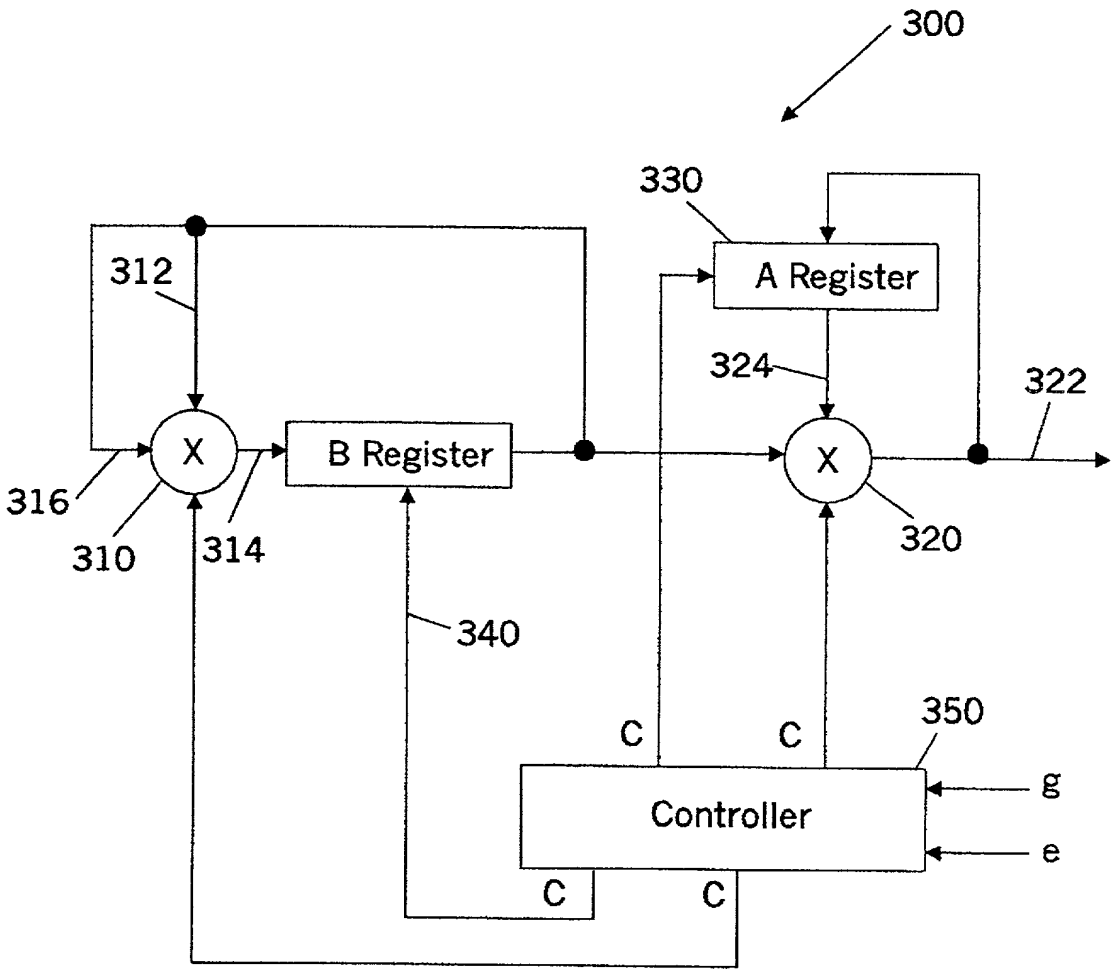

300
330
A Register
312
324
322
X
B Register
X
316
310
314
320
340
C
C
350
Controller
g
e
C
C

FIG. 16

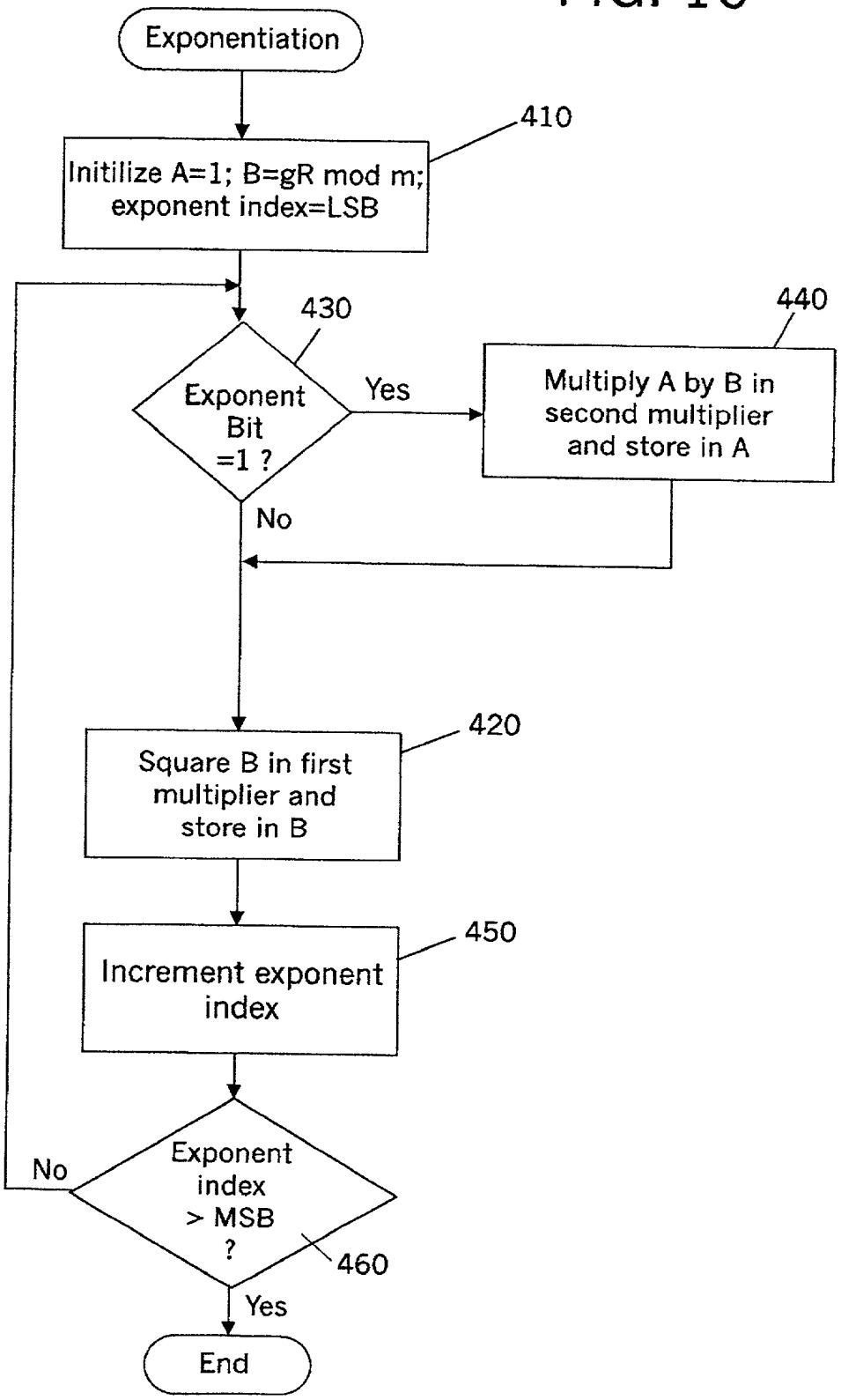

Exponentiation
410
Initilize A=1; B=gR mod m; exponent index=LSB
430
Exponent Bit =1 ?
Yes
440
Multiply A by B in second multiplier and store in A
No
420
Square B in first multiplier and store in B
450
Increment exponent index
Exponent index > MSB ?
No
460
Yes
End

FIG. 17